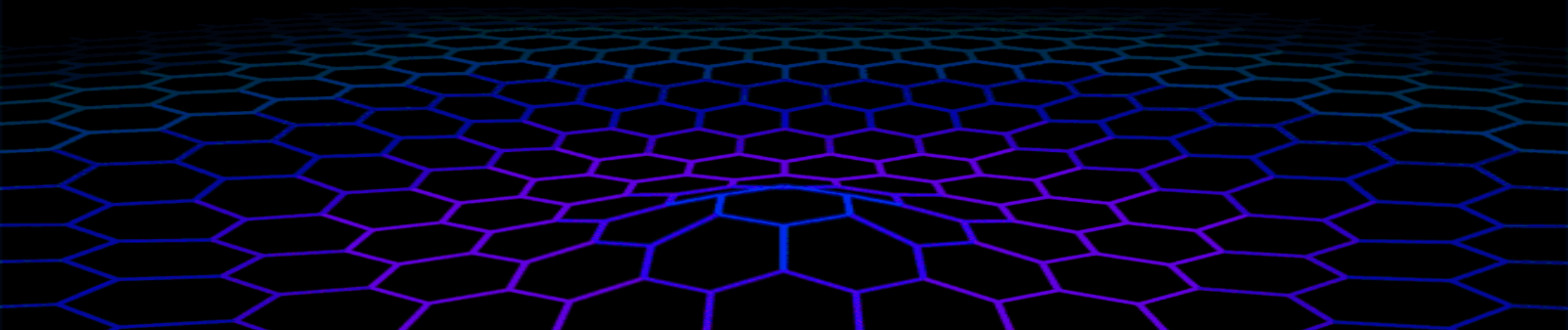
Unisenac
Campus Pelotas

Banco de Dados I




SQL

Resolução dos exercícios da aula anterior (parte 1)

Preparando o terreno

```
CREATE TABLE clientes(
    codigo   CHAR(6),
    nome     VARCHAR(20),
    dataN    DATE,
    valor    DECIMAL(10,2),
    cidade   VARCHAR(20),
    PRIMARY KEY(codigo)
);

-- Posso fazer comentário em uma linha

/* Posso fazer comentário
em mais
de uma linha */
```

Exemplo

- -- Inserindo registros na tabela clientes:

```
INSERT INTO clientes (codigo, nome, dataN, valor, cidade) VALUES ('B5200X', 'Bartolomeu', '1982-12-23', 100.4, 'Pelotas');
INSERT INTO clientes (codigo, nome, dataN, valor, cidade) VALUES ('A73111', 'Orpiliano', '2001-07-19', 98.6, 'Canoas');
INSERT INTO clientes (codigo, nome, dataN, valor, cidade) VALUES ('A77222', 'Dorvalinau', '1999-11-21', 900, 'Pelotas');
INSERT INTO clientes (codigo, nome, dataN, valor, cidade) VALUES ('B79321', 'Bartolomeu', '1969-01-24', 31.29, 'Erechim');
INSERT INTO clientes VALUES ('BX9843', 'Josivaldo', '1967-04-01', 93.45, 'Blumenau');
INSERT INTO clientes (codigo, nome) VALUES ('S9983W', 'Sarafina');
INSERT INTO clientes (cidade, nome, valor, codigo) VALUES ('Pelotas', 'Italinea', 1200.49, 'EL171C');
INSERT INTO clientes VALUES ('XYZ432', 'Anacleto', '1967-04-02', 90, 'Pelotas');
```

```
-- Para visualizar os registros da tabela clientes:
SELECT * FROM clientes;
```

Exemplo

Recuperar todas as colunas da tabela **clientes**.

Recuperar apenas as colunas **nome** e **valor** da tabela **clientes**.

Exemplo

Recuperar todas as colunas da tabela **clientes**.

```
SELECT * FROM clientes;
```

Recuperar apenas as colunas **nome** e **valor** da tabela **clientes**.

Exemplo

Recuperar todas as colunas da tabela **clientes**.

```
SELECT * FROM clientes;
```

Recuperar apenas as colunas **nome** e **valor** da tabela **clientes**.

```
SELECT nome, valor FROM clientes;
```

Exemplo

Recuperar apenas os registros de clientes da cidade **Pelotas**

Recuperar apenas os registros de clientes da cidade **Pelotas** com valor menor do que 100

Recuperar apenas os registros de clientes da cidade **Pelotas ou** da cidade **Erechim**

Recuperar apenas clientes com valor **acima** de **100**.

Recuperar apenas clientes que **não** são de **Pelotas**

Exemplo

Recuperar apenas os registros de clientes da cidade **Pelotas**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas';
```

Recuperar apenas os registros de clientes da cidade **Pelotas** com valor menor do que 100

Recuperar apenas os registros de clientes da cidade **Pelotas ou** da cidade **Erechim**

Recuperar apenas clientes com valor **acima** de **100**.

Recuperar apenas clientes que **não** são de **Pelotas**

Exemplo

Recuperar apenas os registros de clientes da cidade **Pelotas**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas';
```

Recuperar apenas os registros de clientes da cidade **Pelotas** com valor menor do que 100

```
SELECT * FROM clientes WHERE cidade = 'Pelotas' AND valor < 100;
```

Recuperar apenas os registros de clientes da cidade **Pelotas ou** da cidade **Erechim**

Recuperar apenas clientes com valor **acima** de **100**.

Recuperar apenas clientes que **não** são de **Pelotas**

Exemplo

Recuperar apenas os registros de clientes da cidade **Pelotas**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas';
```

Recuperar apenas os registros de clientes da cidade **Pelotas** com valor menor do que 100

```
SELECT * FROM clientes WHERE cidade = 'Pelotas' AND valor < 100;
```

Recuperar apenas os registros de clientes da cidade **Pelotas ou** da cidade **Erechim**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas' OR cidade = 'Erechim';
```

Recuperar apenas clientes com valor **acima** de **100**.

Recuperar apenas clientes que **não** são de **Pelotas**

Exemplo

Recuperar apenas os registros de clientes da cidade **Pelotas**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas';
```

Recuperar apenas os registros de clientes da cidade **Pelotas** com valor menor do que 100

```
SELECT * FROM clientes WHERE cidade = 'Pelotas' AND valor < 100;
```

Recuperar apenas os registros de clientes da cidade **Pelotas ou** da cidade **Erechim**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas' OR cidade = 'Erechim';
```

Recuperar apenas clientes com valor **acima** de **100**.

```
SELECT * FROM clientes WHERE valor > 100;
```

Recuperar apenas clientes que **não** são de **Pelotas**

Exemplo

Recuperar apenas os registros de clientes da cidade **Pelotas**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas';
```

Recuperar apenas os registros de clientes da cidade **Pelotas** com valor menor do que 100

```
SELECT * FROM clientes WHERE cidade = 'Pelotas' AND valor < 100;
```

Recuperar apenas os registros de clientes da cidade **Pelotas ou** da cidade **Erechim**

```
SELECT * FROM clientes WHERE cidade = 'Pelotas' OR cidade = 'Erechim';
```

Recuperar apenas clientes com valor **acima** de **100**.

```
SELECT * FROM clientes WHERE valor > 100;
```

Recuperar apenas clientes que **não** são de **Pelotas**

```
SELECT * FROM clientes WHERE cidade != 'Pelotas';
```

Exemplo

Alterar o nome do cliente **Bartolomeu** para **Bartoloteu**

Alterar a cidade do cliente de código ‘A77222’ para Satolep

Alterar para 90 o valor dos clientes da cidade ‘Pelotas’ que estejam com valor acima de 1000.

Alterar a dataN de todos os clientes para 1º de abril de 2000

Exemplo

Alterar o nome do cliente **Bartolomeu** para **Bartoloteu**

```
UPDATE clientes
   SET nome = 'Bartoloteu'
 WHERE nome = 'Bartolomeu';
```

Alterar a cidade do cliente de código ‘A77222’ para Satolep

Alterar para 90 o valor dos clientes da cidade ‘Pelotas’ que estejam com valor acima de 1000.

Alterar a dataN de todos os clientes para 1º de abril de 2000

Exemplo

Alterar o nome do cliente **Bartolomeu** para **Bartoloteu**

```
UPDATE clientes
   SET nome = 'Bartoloteu'
 WHERE nome = 'Bartolomeu';
```

Alterar a cidade do cliente de código ‘A77222’ para Satolep

```
UPDATE clientes
   SET cidade = 'Satolep'
 WHERE codigo = 'A77222';
```

Alterar para 90 o valor dos clientes da cidade ‘Pelotas’ que estejam com valor acima de 1000.

Alterar a dataN de todos os clientes para 1º de abril de 2000

Exemplo

Alterar o nome do cliente **Bartolomeu** para **Bartoloteu**

```
UPDATE clientes
   SET nome = 'Bartoloteu'
 WHERE nome = 'Bartolomeu';
```

Alterar a cidade do cliente de código ‘A77222’ para Satolep

```
UPDATE clientes
   SET cidade = 'Satolep'
 WHERE codigo = 'A77222';
```

Alterar para 90 o valor dos clientes da cidade ‘Pelotas’ que estejam com valor acima de 1000.

```
UPDATE clientes
   SET valor = 90
 WHERE cidade = 'Pelotas' AND valor > 1000;
```

Alterar a dataN de todos os clientes para 1º de abril de 2000

Exemplo

Alterar o nome do cliente **Bartolomeu** para **Bartoloteu**

```
UPDATE clientes
   SET nome = 'Bartoloteu'
 WHERE nome = 'Bartolomeu';
```

Alterar a cidade do cliente de código ‘A77222’ para Satolep

```
UPDATE clientes
   SET cidade = 'Satolep'
 WHERE codigo = 'A77222';
```

Alterar para 90 o valor dos clientes da cidade ‘Pelotas’ que estejam com valor acima de 1000.

```
UPDATE clientes
   SET valor = 90
 WHERE cidade = 'Pelotas' AND valor > 1000;
```

Alterar a dataN de todos os clientes para 1º de abril de 2000

```
UPDATE clientes
   SET dataN = '2000-04-01';
```

Exemplo

Excluir apenas o cliente que tenha o codigo ‘A73111’.

Excluir todos os registros de clientes.

Exemplo

Excluir apenas o cliente que tenha o codigo ‘A73111’.

```
DELETE FROM clientes
WHERE codigo = 'A73111';
```

Excluir todos os registros de clientes.

Exemplo

Excluir apenas o cliente que tenha o codigo ‘A73111’.

```
DELETE FROM clientes
WHERE codigo = 'A73111';
```

Excluir todos os registros de clientes.

```
DELETE FROM clientes;
```

Resolução dos exercícios da aula anterior (parte 2)

ATIVIDADE - Crie a modelagem lógica e física:

Criar um banco chamado **aula06**

Criar as tabelas:

EMPREGADO (matricula, nome, endereco, salario, matSuperv, codDep)

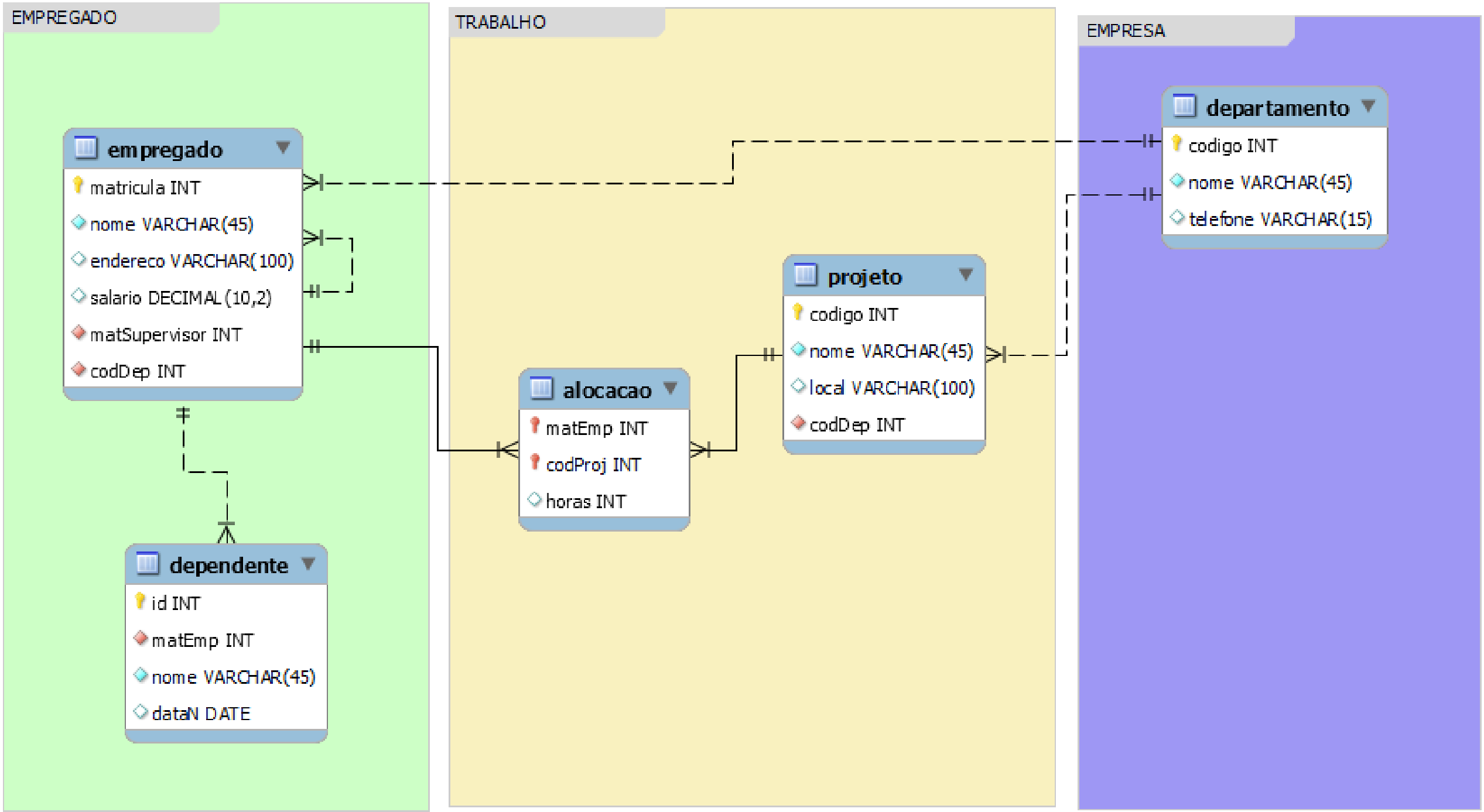
DEPARTAMENTO (codigo, nome, telefone)

PROJETO (codigo, nome, local, codDep)

ALOCACAO (matEmp, codProj, horas)

DEPENDENTE (id, matEmp, nome, dataN)

Solução


EMPREGADO
TRABALHO
EMPRESA
empregado
matricula INT
nome VARCHAR(45)
endereco VARCHAR(100)
salario DECIMAL(10,2)
matSupervisor INT
codDep INT
dependente
id INT
matEmp INT
nome VARCHAR(45)
dataN DATE
alocacao
matEmp INT
codProj INT
horas INT
projeto
codigo INT
nome VARCHAR(45)
local VARCHAR(100)
codDep INT
departamento
codigo INT
nome VARCHAR(45)
telefone VARCHAR(15)

```sql
DROP DATABASE IF EXISTS aula06;
CREATE DATABASE IF NOT EXISTS aula06;
USE aula06;

CREATE TABLE departamento (
     codigo   INT(11),
     nome     VARCHAR(45) NOT NULL,
     telefone VARCHAR(15),
     PRIMARY KEY (codigo)
);
```

```sql
CREATE TABLE empregado (
    matricula     INT(11),
    nome          VARCHAR(45) NOT NULL,
    endereco      VARCHAR(100),
    salario       DECIMAL(10, 2),
    matsupervisor INT(11),
    coddep        INT(11) NOT NULL,
    PRIMARY KEY (matricula),
    FOREIGN KEY (coddep)        REFERENCES departamento (codigo),
    FOREIGN KEY (matsupervisor) REFERENCES empregado (matricula)
);
```

```sql
CREATE TABLE projeto(
     codigo  INT(11),
     nome    VARCHAR(45) NOT NULL,
     local   VARCHAR(100),
     coddep  INT(11) NOT NULL,
     PRIMARY KEY (codigo),
     FOREIGN KEY (coddep) REFERENCES departamento (codigo)
);

CREATE TABLE alocacao(
     matemp  INT(11),
     codproj INT(11),
     horas   INT(11),
     PRIMARY KEY (matemp, codproj),
     FOREIGN KEY (matemp)  REFERENCES empregado (matricula),
     FOREIGN KEY (codproj) REFERENCES projeto (codigo)
);
```

```sql
CREATE TABLE dependente (
    id      INT(11),
    matemp INT(11),
    nome    VARCHAR(45) NOT NULL,
    datan   DATE,
    PRIMARY KEY (id),
    FOREIGN KEY (matemp) REFERENCES empregado (matricula)
);
```

O que é uma Foreign Key no MySQL?

Definição:

Uma Foreign Key (chave estrangeira) é uma coluna (ou conjunto de colunas) que cria uma relação entre duas tabelas.

Ela refere-se à chave primária de outra tabela, estabelecendo uma ligação entre os dados dessas tabelas.

Finalidade:

Garantir a integridade referencial: A Foreign Key assegura que os dados em uma tabela (tabela filha) estejam sempre relacionados aos dados de outra tabela (tabela pai).

Prevenir dados órfãos: Impede que registros sejam inseridos ou atualizados com valores de chaves estrangeiras que não existam na tabela referenciada.

O que é uma Foreign Key no MySQL?

Vantagens do Uso de Foreign Keys:

Integridade de dados: Garante que um registro de uma tabela (como empregado) não possa existir sem um registro correspondente em outra tabela (como departamento).

Evita inconsistências: Se você tentar deletar um departamento que tenha empregados associados, o banco de dados pode bloquear a operação ou automaticamente deletar os registros relacionados, dependendo das opções de restrição definidas.

Foreign Key – Exemplo prático

No caso da tabela empregado, a coluna coddep é uma Foreign Key que referencia a coluna codigo da tabela departamento:

```
FOREIGN KEY (codproj) REFERENCES projeto (codigo)
ON DELETE RESTRICT
ON UPDATE CASCADE;
```

Sintaxe para Criar uma Foreign Key:

```
FOREIGN KEY (coluna_da_tabela_filha)
REFERENCES tabela_pai (coluna_da_tabela_pai)
```

Dica:

Integridade referencial pode ser configurada para impedir a exclusão de registros referenciados (com ON DELETE RESTRICT) ou permitir a exclusão em cascata (com ON DELETE CASCADE), o que também apaga os registros relacionados.

POPULANDO

- Cadastre empregados com salário menor do que 5.000, igual a 5.000 e maior do que 5.000;
- Cadastre pelo menos três funcionários com salário entre 1.700 e 2.800;
- Cadastre pelo menos 10 departamentos (com códigos de 1 até 10);
- Cadastre dois funcionários no departamento 1;
- Cadastre pelo menos três funcionários no departamento 2;
- Cadastre 3 funcionários no departamento 5;
- Cadastre alguns dependentes com data de nascimento igual a 27/10/2002;
- Cadastre alguns dependentes com data de nascimento posterior a 27/10/2002;

Sugestão de conteúdo para a tabela **departamento**.

codigo	nome	telefone
1	RH	321
2	Compras	123
3	Transportes	456
4	Marketing	654
5	Vendas	789
6	Financeiro	987
7	Estoque	234
8	Saúde	432
9	Controladoria Geral	345
10	Ouvidoria	543

```sql
INSERT INTO departamento (codigo, nome, telefone) VALUES (1, "RH", "321");

INSERT INTO departamento (codigo, nome, telefone) VALUES (2, "Compras", "123");
```

Sugestão de conteúdo para a tabela **empregado**.

matricula	nome	endereco	salario	matsupervisor	coddep
800	Josivaldo Antunes Nunes	Rua das flores	4000.00	NULL	1
835	Plinio Cabresto Selvagem	Av Sallus	5000.00	900	3
836	Ortega Raimundo Gomes	Av Marlua	5600.00	900	3
837	Solange Costa Ortiz	Rua Zanzibar	8000.00	835	3
841	Monange Costa Ortiz	Rua Zanzibar	1700.00	NULL	4
842	Rosange Costa Ortiz	Rua Zanzibar	1750.00	NULL	2
843	Violange Costa Ortiz	Rua Zanzibar	2700.00	NULL	5
844	Sustange Costa Ortiz	Rua Zanzibar	2800.00	843	6
845	Zuleiva Maciel Souza	Rua Troll	8050.00	NULL	1
846	Panceta Furunculo Anacleto	Rua Dores	18050.00	844	1
847	Marciano das Antenas Verdes	Rua Marte	1050.00	NULL	2
848	Etevaldo Agusto Moraes	Rua Venus	2050.00	NULL	2
849	Lucrecio Borges Almeida	Rua Jupiter	8049.00	847	2
900	Marzivania Alves Breda	Rua das urtigas	4500.00	NULL	3
947	Zorzicleto Bicicleteiro	Rua Marte	2050.00	NULL	5
948	Pedregusto Mars	Rua Venus	3050.00	NULL	5
949	Laurinda Linda Lindeza	Rua Jupiter	5049.00	847	5

```
INSERT INTO empregado (matricula, nome, endereco, salario, matsupervisor, coddep)
VALUES (800, "Josivaldo Antunes Nunes", "Rua das flores", 4000, NULL, 1);
```

Sugestão de conteúdo para a tabela **dependente**.

id	matemp	nome	datan
1	949	Laurindinha Lindeza	2002-10-27
2	949	Laurindinho Lindeza	2002-10-27
3	947	Bikemotor Yamaha	2002-10-27
4	843	Pringles Doritos de Queijo	2002-10-28
5	848	Fanta Uva de Avila	2004-11-27
6	948	Setembrino Dorvalino	2012-12-26

```sql
INSERT INTO dependente (id, matemp, nome, datan)
VALUES (1, 949, "Laurindinha Lindeza", "2002-10-27");

INSERT INTO dependente (id, matemp, nome, datan)
VALUES (2, 949, "Laurindinho Lindeza", "2002-10-27");
```

Exercícios - Escreva a instrução SQL para:

```
-- 01. Apresentar a listagem completa dos registros da tabela empregado;
-- 02. Apresentar uma listagem dos nomes e salários dos empregados com salario maior
do que 5.000;
-- 03. Listar os nomes e os salários dos empregados em ordem alfabética, decrescente,
de nome;
-- 04. Listar os nomes dos empregados do departamento 5 ordenados pelo endereço;
-- 05. Alterar para 2500,50 o salário dos empregados do departamento 2;
-- 06. Aumentar em 20% o salário de todos os empregados;
-- 07. Excluir todos os empregados do departamento 1;
-- 08. Apresente a listagem dos dependentes que nasceram em 27/10/2002;
-- 09. Apresente a listagem dos dependentes que nasceram após 27/10/2002;
-- 10. Listar os empregados do departamento 5;
-- 11. Listar empregados com salário até 5000,00;
-- 12. Listar empregados com salário entre 1700,00 e 2800,00.
```

Novas questões (aula07)

Exercícios - Escreva a instrução SQL para:

```
-- 13. Listar o nome e a matrícula dos empregados que não possuem supervisor (matsupervisor é
NULL);
-- 14. Alterar o telefone do departamento 'Vendas' para '987654321';
-- 15. Listar todos os empregados cujos salários são maiores que a média dos salários dos
empregados;
-- 16. Listar os nomes e as datas de nascimento dos dependentes em ordem cronológica de
nascimento (do mais antigo para o mais recente);
-- 17. Atualizar o endereço de todos os empregados do departamento de 'Transportes' para
'Endereço Desconhecido';
-- 18. Excluir todos os dependentes com data de nascimento anterior a 2000;
-- 19. Apresentar a quantidade total de empregados em cada departamento;
-- 20. Listar os três empregados com os maiores salários;
-- 21. Atualizar o salário dos empregados com salário inferior a 3000 para 3200;
-- 22. Listar os nomes e salários dos empregados, ordenando primeiramente por salário de forma
decrescente e depois por nome de forma ascendente;
-- 23. Listar os nomes dos dependentes, ordenando primeiro pela data de nascimento de forma
ascendente e, em caso de empate, pelo nome de forma decrescente;
-- 24. Listar os nomes e endereços dos empregados, ordenando primeiro pelo nome de forma
ascendente e depois pelo endereço de forma descendente;
-- 25. Listar os nomes e telefones dos departamentos, ordenando primeiramente pelo telefone de
forma ascendente e depois pelo nome de forma descendente;
```